Design Consideration for Client Synchronization Methods in an Identity Resolution Service

Fumiko Kobayashi, John R. Talburt
Department of Information Science
University of Arkansas at Little Rock
2801 South University Ave. EIT 550
Little Rock, AR, USA

Abstract- Entity identity information management (EIIM) systems provide the information technology support for master data management (MDM) systems. One of the most important configurations of an EIIM system is identity resolution. In an identity resolution configuration, the EIIM system accepts entity identity information and returns the corresponding entity identifier. In the original EIIM model, identity resolution was only a batch operation. After the model was extended with an Identity Resolution Service (IRS) [1] to decouple identity resolution from batch EIIM processing, identity resolution moved into the interactive realm with additional functionality. This paper discusses the design consideration for one of these additional functions specifically centered on synchronization of entity identity information amongst client systems. This paper outlines the need and importance of client synchronization along with a design for obtaining client synchronization.

Keywords- Identity Resolution Service; Entity Identity Information Management; Syncronization; Master Data Management; Entity Resolution

1 Introduction

Identity resolution (IR) is the process of determining if an entity reference refers to the same entity as one of the entity identity structures (EIS) under management in an entity identity information management (EIIM) system. IR is sometimes called “entity recognition” because the system is being asked if the input entity reference can be recognized as one of the entities already under management.

ER is the process of determining whether two references to real-world objects in an information system are referring to the same object, or to different objects [2]. Real-world objects are identified by their attribute similarity and relationships with other entities. Some examples of attributes for person entities are First Name, Last Name, and Social Security Number (SSN). For place or location entities the attributes might be Latitude, Longitude, Description, or postal address. ER has also been studied under other names including but not limited to record linkage [3], deduplication [4], reference reconciliation [5], and object identification [6].

Previous IQ research [8] [9] has extended ER into the larger context of entity identity information management (EIIM) that includes the creation and maintenance of persistent data structures to represent the identities of external entities [2]. The overall goal of EIIM is to allow the ER system to achieve entity identity integrity, a state in which two conditions hold [10].

1. Each identity structures corresponds to one, and only one, real-world entity
2. Distinct identity structures correspond to distinct real-world entities.

Entity identity integrity is another way of stating the Fundamental Law of Entity Resolution [2] which requires that two entity references should be linked if, and only if, they are equivalent where equivalence means both reference the same real-world entity.

In the current model of EIIM, the configurations to maintain the EIS operate primarily in an offline batch mode. In general the EIIM model is focused on the processes necessary to achieve and maintain entity identity integrity of the EIS under management in systems identity knowledgebase (IKB). EIIM provides the tools to support the complete lifecycle of identity information.
2 Problem Definition

The current EIIM model simply assumes that some external clients systems are providing entity identity information to the EIIM process, and in turn, the EIIM process is providing corresponding entity identifiers in the form of the link index table as described previously. However, no provision is made as to how these systems use and manage these entity identifiers. In particular, the issue of synchronization (consistency) is not addressed in the EIIM model. The problem is that when more than one system provides identity information to the EIIM system, identities in these systems can be inconsistently represented. For example, if System A provides an input record that is assigned entity identifier “345”. Later System B provides new information causes the EIS assigned entity identifier “345” to be merged into the EIS with entity identifier “123”. In this case, the identity “345” is no longer a valid entity identifier in the IKB; it has been superseded by entity identifier “123”. This change will be reflected in System B that provided the information that caused the change, but System A is still using the entity identifier “345” for the same identity. This is illustrated in Figure 1.

There is no formal model for identity life cycle management which encompasses the client synchronization of EIIM systems to enforce persistent identification across various systems. Loss of client synchronization occurs when entity identifiers in the IKB change as the EIIM system performs updates to the centralized repository.

3 Identity Resolution Service (IRS)

An extension or new layer of functionality is required for the existing EIIM system. This new extended system is referred to as the Identity Resolution Service (IRS). Figure 2 shows that the additional layer added to an EIIM system creates the IRS that incorporates and maintains the functions of the EIIM system itself.

The IRS can be invoked by applications from multiple remote client systems allowing them to retrieve identity interactively [1] [11]. By developing an IRS, clients are able to decouple and separate their identity management rules and processes from other business rules and processes.

The interactive IR system operates by accepting a reference(s) from the client system and then resolving the reference against the IKB. It accepts and returns one or more references depending on the mode of operation. If the IR system can identify matches, then the system returns the relevant identifiers to the client (along with other information depending on configuration). If the system is unable to find a positive match, then the system returns the most likely match along with a confidence rating. This confidence factor is calculated through the use of a probabilistic score algorithm [12] [13]. The confidence factor can help the client make a decision as to which possible match is the actual match (if any). If neither a match nor a set of possible matches can be identified, the system should inform the client or return an empty list to the client.
With the relocation of IR to the IRS layer, it is still tightly coupled with ER processing but allows for a different type of processing of the data. This IRS layer improves the IQ of the information by allowing it to be retrieved in a timely manner; by accessing the most up to date IKB so the data is not stale, and will even make it easier for users to query the data which provides them a higher likelihood of identifying issues with the data and correcting it. Figure 3 shows the high-level components of an IRS.

By introducing interactive identity resolution and a client synchronization framework, the IRS extends and enhances the EIIM system to fill the void that was left in regards to design for the Resolve and Retrieve Phase of the CSRUD (capture, store and share, resolve and retrieve, update, and dispose) MDM life cycle. For practical use, the Resolve and Retrieve Phase is the most important of all the of the CSRUD MDM life cycle phases. Resolving an entity reference to its correct entity (EIS) is the primary use case for MDM. It’s this resolve and retrieval that provides actual value to the client systems.

A major issue for Resolve and Retrieve Phase is the client synchronization of entity identifiers in the centralized IRS system with entity identifiers residing in client systems. As entity identifiers change in the IRS IKB, the changes must be propagated to the clients’ systems. This paper focuses on the need for and the design of the client synchronization functionality of an IRS.

### 4 Client Synchronization

Although there is no research directly addressing the identity synchronization in ER systems, the concept of synchronization has been studied for many other uses in the Information technology field. Data synchronization is the process of establishing consistency among data from a source to target data storage and vice versa. This consistency must be maintained between sources over time. Data synchronization is considered one of the key concepts behind most computing systems and networking protocols available. Data synchronization appears in many research fields such as distributed systems, mobile database [14], pervasive computing [15], mobile learning [16], data grids [17], and peer-to-peer content distribution [18].
When performing client synchronization of the EIIM systems, the proper design of the client synchronization procedures is vital in insuring the correctness of the identities in all effected EIIM systems. Correctness of resolution is the main objective in ER systems and this importance must be carried through to synchronized EIIM systems.

Before design for the client synchronization methods can be done, the overall goal of an interactive system must be considered. One of the primary objectives in an interactive system is the timeliness of responses. By adding client synchronization methods to an interactive system, no impact to processing time should be experienced. To facilitate this, the use of logging is integrated into the client synchronization design. The final design encompasses two types of client synchronization, push and pull [19].

By including both models, it provides more robust methods of client synchronization to accommodate a wider variety of client systems.

4.1 Logging

The client synchronization design requires two types of logging, a change log and a request log.

The change log stores a list of all entity identifier values that have changed in the IKB along with the new entity identifiers value that they have changed to. These changes occur when a structure-to-structure assertion [20] run is performed by the EIIM system to bring multiple EIS into a single EIS. This also occurs when an input source references is found to be a "glue record" that brings multiple EIS together in the IKB. It is important to log these changes as it allows for accurate client synchronization of the IRS with remote client systems.

The request log stores a list of all the entity identifiers that have been sent to a client system along with ID associated to the client system.

4.2 Push Client Synchronization

In a push model, outlined in Figure 4, an update to the Change Log signifies a change to an EIS ID. When this occurs, the notification system queries the Request Log looking for any previous client systems that have requested or were returned the changed EIS ID. If a client system can be identified then the notification system will “Push” the updated information to the client system so that they can integrate it into their system and avoid having stale information. Figure 5 provides a more detailed example.

In this scenario the following occurred:

- An update occurred to the IKB that caused EIS E to merge with C
- The push notification logic noticed the new addition to the Change Log
- The IRS searched the Request Log for any request that had previously been provided ID E.
- Client system 1 was identified as having been provided this ID in the past
- The IRS system sends a push notice to Client system 1 alerting them to the update of ID

It is the responsibilities of the Client system to have a system in place that can receive and process these push notifications.

![Figure 5: Push Client Synchronization Example](image)

For the push model of client synchronization to function correctly, some sort of “registration” must occur between each client system and the IRS. This is done via registration to the IRS and the assignment of an access key to the client. This works by allowing a client to register their system with the IRS and predefine the method for which they wish to receive notifications into their system. Once registration is complete the client will be assigned an access key which they must supply for every subsequent IR request into the system. The entity identifier that the client retrieved from the IRS is logged into the Request Log along with their access key. When an entity identifier is changed in the IKB, the Request log and access key are used to pick the correct connection and method for which to send the change notification.

For a fully optioned push client synchronization, two methods of notification are proposed. The first is the use of e-mail notifications. When the client system registers to receive e-mail notifications for changes, they must specify an e-mail address this is in turn associated with their access key in the IRS. When a change to an entity identifier occurs, an e-mail containing the following body is sent:

```
{OldID=XXXXXXXXXX;NewID=YYYYYYYYY}
```

The client can parse the e-mails and update their local IKB.

The second method of push client synchronization is done via an API call. The client system must have an API for which
the IRS can make a POST call. The body of the call will include an XML structure that conveys the change. This will look like this:

```xml
<Change_Notification>
  <OldID>XXXXXXX</OldID>
  <NewID>YYYYYYYY</NewID>
</Change_Notification>
```

The client system must be able to receive and process the xml notification posted to their API. The client must provide the web address of their API when registering for an access key in the IRS.

### 4.3 Pull Client Synchronization

In a pull model, depicted in Figure 4, the Client system submits an EIS IDs that was previously provided to them by the IRS service back to the system. The notification system will query the Change Log and identify any changes that occurred for the provided EIS ID. If a change is found, this information is provided to the Client system. By having the Logs in conjunction with the notification framework, it is possible to provide Client systems with relevant information beyond what is possible using the IKB alone. Figure 6 provides an example of how this notification configuration functions in the IRS.

In this scenario the following occurred:

- The Client system submitted a request to check if ID for EIS E has changed
- The pull notification logic searches the Change Log
- It finds that EIS E has changed and merged with EIS C
- The IRS returns this change to the client system.

This may seem similar to how the ID Search feature works but with one significant different. This request requires much less processing as the in-memory IKB is never touched to find the answer. Only a search against the smaller indexed change log is required. If the client system Later decides they would like the full content of the new EIS, they will have to submit a new ID Search request.

In a pull synchronization, no access key is required. This is a process that is strictly initiated from an API call from the client system. The client sends an entity identifier they have in their IKB and the system sends them back the following if a change to the identifier is located:

```
{OldID=XXXXXXXXXX;NewID=YYYYYYYYY}
```

If no change is identified, the system will return a null set like:

```
{NULL}
```

### 5 Summery

The design of a client synchronization and notification framework extends the entity identity life cycle management model to allow for client synchronization of entity identifiers across client systems.

The separation of IR from the EIIM system to improve access to the information stored in the identities creates the problem of lost client synchronization. A logical extension to the EIIM model is a client synchronization maintenance task to allow for the information received by clients to remain current if the data in the Knowledgebase is updated. This client synchronization is designed for inclusion into the IRS.

The client synchronization services, used by Client 1 in Figure 5 and Figure 6, provide a mechanism for the clients to keep the freshest identifiers for their identities and stay in sync with the centralized IKB. By including both models for client synchronization, push and pull, in the system.

In a more general sense a pull model becomes the users’ responsibility to inquire into the management system to determine if change has occurred or not to their identities in their local IKB. A pull notification would provide applications developers the ability to send an identifier that is in use in their system and check if it is still in sync with the centralized repository without having to perform resolution on the reference again.

One possible model for the client synchronization service built as a push notification would be a feature of the Identity Service which pushes a combination of old identifier and new identifier to clients that have made requests to the system in the past. It is up to the client to process the update notification when it is received and update their information. The push model is a more complex model which places the responsibility on the management system to remember which clients, and which client references have been processed and to log changes and push the changes back to the client system.
No matter which model is selected for use, the benefits of a client synchronization framework are clear in that it solve the issue of mismatched identity state that exists in the current EIIM model. It also adds value to the IRS and improves the accuracy of the data stored in the client IKB since there is a method to address and update stale data.

6 Future Work

Future work for a client synchronization framework will consist of extensive testing of the frameworks proposed in this paper. This experimentation will be aimed at providing validation that the proposed models are viable for a production system. Issues surrounding the concepts of commit and rollback of entity identifier synchronization must also be addressed.

Acknowledgment

The research described in this paper has been supported in part by funding from the Arkansas Department of Education.

References


